

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CISCO SYSTEMS, INC.,
Petitioner

v.

QUICKER CONNECTIONS LLC,
Patent Owner

IPR2025-01088
U.S. Patent No. 7,061,859

**DECLARATION OF DR. NENAD MEDVIDOVIĆ IN SUPPORT OF
PATENT OWNER'S PRELIMINARY RESPONSE**

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	QUALIFICATIONS	1
III.	BASIS OF OPINIONS	9
IV.	LEGAL PRINCIPLES.....	9
V.	LEVEL OF ORDINARY SKILL IN THE ART	10
VI.	CLAIM CONSTRUCTION	11
	<i>A. Mask</i>	11
	<i>B. Null</i>	12
	<i>C. Superimposing</i>	12
VII.	THE '859 PATENT	15
	A. Overview of the '859 Patent	15
	B. The '859 Patent's General Mask.....	20
VIII.	THE PRIOR ART - OMICHI.....	22
IX.	THE CHALLENGED CLAIMS ARE NOT OBVIOUS IN VIEW OF OMICHI.....	24
	A. The Petition Fails to Teach, Suggest, or Render Obvious the Claimed "General Mask"	24
	1. Omichi discloses only a link failure table is not the '859 Patent's link-reachability general mask	26
	2. A POSITA would not be motivated to modify Omichi to include the '859 Patent's claimed general mask.....	27
	B. Omichi Fails to Teach, Suggest, or Render Obvious the Claimed "Superimposing"	29

1. Only Bitwise Boolean operations, rather than serial logical Boolean operations, would meet the “superimposing” claim element	29
2. The claimed “superimposition” (i.e. bitwise Boolean operations) are not taught or suggested by Omichi and instead such disclosures are conspicuously absent.....	34
C. Omichi Does not Disclose or Render Obvious not Conveying a Data Flow when the Destination Node is Unreachable [Claims 3, 6, 9, 12]...	40
X. CONCLUSION.....	43

I, Nenad Medvidović, of Los Angeles, California declare that:

I. INTRODUCTION

2. I have been retained by counsel for Patent Owner Quicker Connections LLC (“Quicker Connections,” “QC,” or “Patent Owner”) as an independent expert in the *inter partes* review (“IPR”) proceedings involving U.S. Patent No. 7,061,859 (the ’859 Patent). have been asked to consider the arguments raised by the Petition regarding the validity of the ’859 Patent in this proceeding.

3. It is my understanding that the sole asserted ground of invalidity is obviousness of all claims (1-12) of the ’859 Patent in view of Japanese Patent Publication No. JP2001053772A to Omichi (“Omichi”).

Ground	Reference	Basis	Challenged claims
1	Omichi	Obviousness	1-12

4. I am being compensated for my work in this matter at my usual hourly rate for this work. I am also being reimbursed for reasonable and customary expenses associated with my work and testimony in this investigation. My compensation is not contingent on the outcome of this matter or the specifics of my testimony.

II. QUALIFICATIONS

5. My qualifications as an expert are detailed in my curriculum vitae, a copy of which may be found in Exhibit 2059.

6. I received a Bachelor of Science degree in Computer Science, *summa*

cum laude, from Arizona State University's Computer Science and Engineering department.

7. I received a Master of Science degree in Information and Computer Science from the University of California at Irvine's Information and Computer Science department.

8. I received a Doctor of Philosophy degree from the University of California at Irvine's Information and Computer Science department. My dissertation was entitled "Architecture-Based Specification-Time Software Evolution."

9. For over thirty years, my research has focused on software systems; their design, development, maintenance, and evolution; their individual building blocks and overall architectures; and mechanisms for their implementation and deployment. My Ph.D. dissertation yielded a new design style for the development and evolution of highly distributed software systems built out of independent components that exchange different types of information encoded in messages to accomplish specific goals. This research focused on the design principles behind those systems, the appropriate support for modeling the systems (both at the level of individual modules as well as their integrations into subsystems and entire systems), and the infrastructure for their implementation, deployment, optimization, and long-term evolution. This work was applied across a wide range of domains, from

traditional distributed systems, to mobile devices and applications, to embedded systems and autonomous robots.

10. My work has been funded by US government agencies (e.g., National Science Foundation, National Institutes of Health, Defense Advanced Research Projects Agency, NASA, Department of Homeland Security) and industrial organizations (e.g., Boeing, IBM, Northrop Grumman, Xerox, Google, Bosch).

11. I am employed by the University of Southern California (“USC”) as a faculty member in the Computer Science Department and have been since January 1999. I am a tenured professor. Since July 2022, I have served as my Department’s Chair. Between January 2009 and January 2013, I served as the Director of USC’s Center for Systems and Software Engineering (USC-CSSE), a large research center that involved the work of a number of USC faculty, students, and staff as well as several dozen industry partners. Under my direction, USC-CSSE was involved in a range of cutting-edge projects for our industry partners, ranging from the Aerospace Corporation to Xerox. Between July 2011 and July 2015, I served as my Department’s Associate Chair for Ph.D. Affairs. Between July 2021 and June 2022, I served as my Department’s Vice Chair for Tenure Affairs. In July 2025, I was named TRW Professor in Engineering and given an endowed chair for my contributions to my university and the academic community.

12. Based on my experience and expertise, in 2007 I was offered the

position of Chief Software Engineer of the Transformational Satellite Communications System (TSAT) program. This was a \$4B U.S. Department of Defense (DOD) program, sponsored by the U.S. Air Force. The program's charge was to build the satellite grid of the future. With my university's agreement, I was prepared to go on a multi-year leave-of-absence assignment until the TSAT project was operational. Unfortunately, the program's funding was cut before my involvement could begin. In July 2025, I was named TRW Professor in Engineering and given an endowed chair for my contributions to my university and the academic community.

13. I have created and/or taught undergraduate-, Master's-, and PhD-level courses in software engineering, software requirements, software architecture, and embedded systems for many years. As part of these courses, I have developed and taught modules, and created and guided projects that exposed students to the interplay between software and traditional computing devices on one hand, and a wide range of physical equipment on the other hand, including sensors, RFID systems and their integration into item tagging applications, cameras, mobile robots, and wearable computing devices.

14. I have many years of extensive experience with wireless communications and network communications including, by way of example only, the RFID systems mentioned above and the Transformational Satellite

Communications System mentioned above.

15. In the early 2000s, I secured a grant from the Intel Corp., which I used to equip a teaching laboratory with a variety of embeddable computing equipment. These are non-traditional computing platforms, such as those listed above, that can be a part of devices used in everyday activities (e.g., “smart” cards, cameras, phones, clothing accessories, etc.). This laboratory was used as the foundation of a graduate-level course, Software Engineering for Embedded Systems, that I introduced soon thereafter and taught a number of times. This course covered a range of topics relevant to embedded systems, including efficient, reliable, and secure communication in networks of varying sizes and the challenges inherent therein. Every semester, my students had to complete a course project they selected from a range of real-world scenarios. These projects included real-time image tracking and recognition; incremental automated scene discovery and rendering; efficient and secure off-board data compression, storage, and access; and so on. Several of these projects grew into peer-reviewed publications and doctoral dissertations. Subsequent extensions of the projects from this course have been used externally, at NASA’s Jet Propulsion Laboratory and Bosch Research and Technology Center.

16. I have many years of extensive experience with network communications including, by way of example only, the RFID systems mentioned above and the Transformational Satellite Communications System mentioned above.

17. A significant portion of my professional activity centers on conducting research through advising Ph.D. students. I have graduated 21 Ph.D. students to date, and I advise seven students currently pursuing their Ph.D. My students have gone on to successful careers in top software companies (e.g., Google, NVIDIA, Intel, Snap), research institutes (e.g., NASA's Jet Propulsion Laboratory, Information Sciences Institute, Quandary Peak Research), and universities (e.g., University of Massachusetts, University of California, University of Edinburgh in the UK).

18. I served as Program Co-Chair for the flagship conference in my field—International Conference on Software Engineering (“ICSE”)—held in May 2011. I have served as Chair or Co-Chair for various other conferences in the Software Engineering field, including: the Fifth Working Institute of Electrical and Electronics Engineers (“IEEE”)/IFIP Conference on Software Architecture, the Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems, the Fifteenth International Association for Computing Machinery’s (“ACM”) Special Interest Group on Software Engineering (“SIGSOFT”) Symposium on Component Based Software Engineering, the IEEE/CSSE/ISE Workshop on Software Architecture Challenges for the 21st Century, and the Doctoral Symposia at the Sixteenth ACM SIGSOFT International Symposium on the Foundations of Software Engineering and at the 44th International Conference on Software Engineering.

19. I have served as an editor of several peer-reviewed journals, including IEEE Transactions on Software Engineering, ACM Transactions on Software Engineering and Methodology, Journal of Software Engineering for Robotics, Elsevier Information and Software Technology Journal, Journal of Systems and Software, Journal of Software Engineering Research and Development, and Springer Computing Journal. I have also served as a guest editor of several special issues for other journals. Between January 2018 and December 2022, I served as Editor-in-Chief of IEEE Transactions on Software Engineering (“TSE”), the flagship software engineering research journal.

20. Between September 2013 and September 2015, I served as Chair of the ICSE Steering Committee. Both before that (2008-2013) and subsequent to that (2015-present) I have been a member of the ICSE Steering committee. I am currently a member of the Steering Committee of the joint European Conference on Software Engineering and the Symposium on the Foundations of Software Engineering (“ESEC/FSE”). From July 2015 to September 2017, I chaired the Steering Committee of FSE, while it was a standalone symposium. I have served as a member of the Steering Committees of the Working IEEE/IFIP Conference on Software Architecture on two separate occasions, including presently.

21. Between July 2015 and June 2018, I served as Chair of the ACM SIGSOFT, the largest professional organization in my field. During the past year, I

have chaired the committee in charge of ACM's Software System Award, one of the most important technical awards in my field.

22. I co-authored *Software Architecture: Foundations, Theory, and Practice*, a widely used textbook in software systems architecture.

23. I have served as editor of various books in the Software Engineering field, including *Proceedings of the 3rd International Conference on Self-Adaptive and Self-Organizing Systems*, *Proceedings of the Warm-Up Workshop for the 32nd International Conference on Software Engineering*, *Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture*, and 40 Editions of *ICSE*—the Ruby Anniversary Celebration.

24. I have authored or co-authored nearly 300 papers in the Software Engineering field. My work has been cited over 21,000 times. My most cited paper alone has been cited nearly 3,200 times; that paper recently received a Most Influential Paper Award from IEEE TSE. A paper I co-authored in the 1998 ICSE, my field's flagship conference, was given ten years later, in 2008, that conference's Most Influential Paper Award. Another of my publications received a Most Influential Paper Award in 2020. Several of my other publications have received "best paper" or "most cited" awards.

25. My research frequently results in the development of software tools that accompany and demonstrate the published ideas. A paper describing a security-

vulnerability detection and remediation tool for Android devices built in my lab, won the Best Tool Paper award at the 2017 Automated Software Engineering Conference. That idea was subsequently awarded a US patent. I am a co-inventor on two additional issued patents in distributed systems design, implementation, deployment, and computer security.

26. I have been named a Distinguished Scientist of the ACM. I have been elected a Fellow of the IEEE. Fellow of the IEEE is IEEE's highest grade; less than 0.1% of its membership earns this grade each year.

III. BASIS OF OPINIONS

27. My opinions in this Declaration are based on my review of the '859 Patent, its prosecution histories before the U.S. Patent and Trademark Office, the IPR Petition concerning it, and references cited therein in support of the Petition, as well as the references cited in the Patent Owner's preliminary response. I also draw upon my experience and knowledge of computer networking, software engineering, and other experience relevant to the subject matter of the '859 Patent.

IV. LEGAL PRINCIPLES

28. Although I am not an attorney, I have had certain aspects of patent law explained to me by counsel. I have relied on and applied my understanding of such principles in my analysis.

29. I have been informed that establishing that a patent is considered to be

obvious under 35 U.S.C. § 103 requires showing that the differences between the challenged claim and the prior art are so insignificant that the subject matter, as a whole, would have been obvious to a POSITA at the time of the invention.

30. I further understand that obviousness is considered on the basis of each individual challenged claim and that a prior art combination must teach all limitations of a challenged claim in order to render that claim obvious.

31. I understand that the determination of obviousness is based on several underlying factors, including: (1) the scope and content of the prior art; (2) the differences between the claim and the prior art; (3) the level of ordinary skill in the art; and (4) any objective evidence of nonobviousness.

32. I understand that a dependent claim incorporates by reference all limitations of the claim from which it depends.

V. LEVEL OF ORDINARY SKILL IN THE ART

33. In my opinion, a person of ordinary skill in the art (POSITA) would possess at least a bachelor's degree in computer science, electrical engineering, or equivalent training with an additional 2-4 years' experience in networking technologies or equivalent field. Additional work experience could substitute for education, or vice versa.

34. I would have been a POSITA at the time of the '859 Patent's filing date in August, 2001.

35. When considering the alleged prior art, I thought about the students and colleagues I work with that have the skills of a person having ordinary skill in the art and tried to put myself in their shoes. My opinions are based on my understanding of the knowledge of a POSITA at the time of the invention unless otherwise noted.

VI. CLAIM CONSTRUCTION

36. I understand that, in an IPR, claim terms are interpreted in light of the specification and prosecution history, as interpreted by a POSITA. I also understand that claim terms are construed in accordance with their ordinary and customary meaning as would have been understood by a POSITA at the time of the invention, unless the patent applicant or inventor has set forth a special meaning for a term.

37. I have reviewed Petitioner's proposed constructions of "mask" and "null" and largely agree with them, although I would adjust the wording of the proposed construction of "mask" to from "a pattern of characters that is used to control the retention or elimination of portions of another pattern of characters" to "a pattern of **bits** that is used to control the retention or elimination of portions of another pattern of **bits**."

A. Mask

38. It is clear from the '859 Pattern that the claimed masks are sequences of bits. For example, '859 Patent claim 1 refers to "constructing a general mask indicating which of the segments are reachable in the event of a fault in the network

by assigning a respective general mask bit to each segment, indicating whether the segment is reachable following occurrence of the fault.”

39. Additionally, the use of “mask” to refer specifically to bit sequences (for example, subnet masks) would have been familiar to a POSITA at the time of the invention, whereas a more general usage encompassing “characters” would have been atypical.

40. Accordingly, I believe the correct construction of the term “mask” in the ’859 Patent should be slightly revised from “a pattern of characters that is used to control the retention or elimination of portions of another pattern of characters” to “a pattern of **bits** that is used to control the retention or elimination of portions of another pattern of **bits**.”

B. Null

41. While I do not believe that an express construction of the term “null,” is needed, I agree with the Petition and Dr. Min that “null” means a zero value. (*See* Ex. 1003, Min Declaration, at ¶ 47, Petition at 16).

C. Superimposing

42. I believe that the term “superimposing” in the ’859 Patent should be construed to mean “as part of a routing determination, combining or comparing by overlaying all bits as a whole.”

43. A POSITA would understand “superimposing” in the context of bit

vectors and masks (i.e., the context of the '859 Patent) to refer to aligning a mask with a bit vector of equal length and performing a bitwise binary¹ operation (e.g., binary AND, binary OR, or binary XOR) to yield a new result in view of the mask. This is consistent with, for example, the construction of “mask” (“a pattern of bits that is used to control the retention or elimination of portions of another pattern of bits.”)²

44. A POSITA would understand that bit vectors of equal length would have to be compared *in their entirety* in order for masking operations to be effectuated – in particular, a mask dictates which bits of the bit vector to which it is compared are retained or eliminated, and so it must be aligned to the bit vector to which it is compared. In the context of the '859 Patent, the general mask as a whole would have to be compared to specific mask as a whole in order to ensure that the bits corresponding to the links of the desired route (reflected in the specific mask) were properly compared to the link reachability bits of the general mask.

¹ “Binary operation” referring to taking two arguments rather than to operating on zeroes and ones. Bitwise AND, OR, and XOR are examples of bitwise binary operations (typically written as “&”, “|” and “^” respectively in source code). A bitwise *unary* operation would be, e.g., flipping all the bits in a bit vector (bitwise NOT, typically written as tilde, (“~”) in source code, i.e. setting all zeroes of a bit vector to ones and vice versa).

² While the Petition’s proposed construction uses “characters” rather than “bits,” I do not understand this difference to reflect a material difference of opinion between myself and the Petition or Dr. Min’s understanding, as no characters other than bits appear to be at issue with respect to the Petition or the '859 Patent.

Preferably, constructing the general mask includes assigning a respective general mask bit to each of the segments, indicating whether the segment can be reached following occurrence of the fault. Most preferably, constructing the specific mask includes assigning a respective specific mask bit to each of the segments, indicating whether the flow was to be conveyed over the segment had the fault not occurred”

(Ex. 1001, '859 Patent at 2:62-67.)

“For the purposes of this method, each node builds and maintains two masks—a general mask and a specific mask preferably **in an ordered bitmap form**, with one bit for each of segments 31-37.”

(Ex. 1001 at 4:42-45) (emphasis added).

A POSITA would understand that the description of the output of superimposing the masks would only be meaningful if the masks were aligned and overlaid in their respective entirety: *Cf.* Ex. 1001 at 5:8-13:

In the situation shown in FIG. 3, however, the conjunction of the masks is 0000100, which is equal to the general mask itself. In this case, the node concludes that the current data flow overlaps all of the unreachable segments, and therefore steers the data flow around the ring in the opposite direction, at a steering step 58, as shown in the figure.

45. This understanding is consistent with both the claims of the '859 Patent (*See, e.g.* Cl. 2, “A method according to claim 1, wherein superimposing the masks comprises combining the masks by means of a Boolean operation.”) and with what I understand to be the position of the Petition and Dr. Min. *See, e.g.* Petition at 40 (annotated figure of bitwise AND operation showing comparison of complete aligned bit strings); Min Decl. at ¶ 81 (same figure relied on for “superimposing”

limitation).

46. Because the '859 Patent appears to contemplate that other routing decisions be made as part of the superimposition step, (*See, e.g.* Cl. 3, (“A method according to claim 1, wherein Superimposing the masks comprises determining whether to convey the data flow over the desired path, to steer the data flow.”)), a POSITA would understand the overlaying of the bits as a whole to be a *necessary* part of the claimed superimposing, but not necessarily always *all* of that claim element, as the inventor is using the word to refer to both the superimposing *per se* (i.e. the comparison of the bits as a whole) and the routing determinations made on that basis. Thus, in my opinion the term “superimposing” should be construed to mean “**as part of a routing determination**, combining or comparing by overlaying all bits as a whole.”

VII. THE '859 PATENT

A. Overview of the '859 Patent

47. The '859 Patent relates to methods and devices for packet routing in response to fault detection in bidirectional ring networks – i.e., networks in which (i) each node is connected to exactly one node on either side, forming a closed loop, and (ii) in which data can flow in either direction (i.e., either clockwise or counterclockwise).

FIG. 1

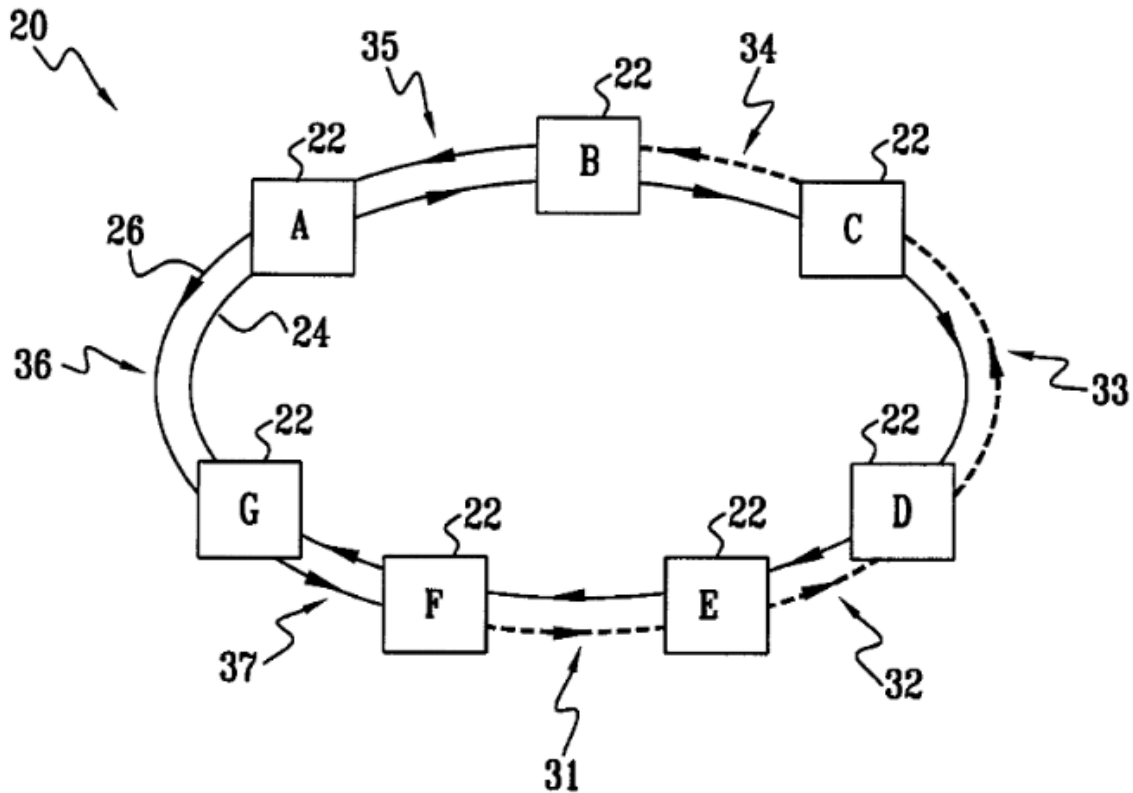


FIG. 1 of the '859 Patent – an example of a bidirectional ring network topology. (Ex. 1001 FIG 1.)

48. “Nodes” (letters A-G in FIG 1 above) generally refers to switching equipment either connected to computers (“access nodes”) or else serving as a gateway to other networks (“gateway nodes”) in the ring network topology. Ex. 1001 at 3:55-57 (“The nodes typically comprise switching equipment, and may serve as either gateways to other networks (aggregation points) or access points.”) 4:7-18 (“FIG. 2 is a block diagram that schematically shows details of one of nodes 22, in accordance with a preferred embodiment of the present invention. Typically, the

node comprises a pair of MAC blocks 40 and 42, with block 40 serving inner ring 24, and block 42 serving outer ring 26. MAC blocks 40 and 42 are coupled to and controlled by a network processor 44. The network processor preferably comprises a chip or chip set that is able to access and process higher-layer protocol information. It performs network-layer functions, Such as IP processing and routing of data flows, as well as delivery of packets to users (in the case of access n)").

49. "Links" refers to bidirectional segments connecting nodes (31-37 in FIG. 1). Ex. 1001 at 3:50-56 ("FIG. 1 is a block diagram that schematically illustrates a bidirectional ring network 20, in accordance with a preferred embodiment of the present invention. Network 20 comprises a plurality of nodes 22, labeled A through [G]³ which are mutually connected by network segments 31, 32, 33, 34,35, 36 and 37."); *id.* at 57-59 ("The network segments comprise a bidirectional communication medium, such as optical fibers or conductive wires.").

50. In ring networks like those of the '859 Patent, communication between any two nodes X and Y can take one of two alternate paths (clockwise or counterclockwise), with traffic routed through intermediate nodes if X and Y are not adjacent to each other on the chosen path.

51. Because traffic can flow between either of two paths between a first

³ The '859 Patent specification reads "A through C" rather than "A through G" here in a way that is an obvious typographical error.

node X and a second node Y, in the event of a link failure that does not disconnect both possible routes from X to Y traffic can be rerouted from a first (default) direction that would traverse the failed link to an alternative path going the other direction around the ring. If each node is aware of all link failures, then each node will also have the information to route packets appropriately to take the non-failed path (if one exists) to a given node or, if no such path exists (due to multiple link failures, at least one in each direction) the origin node can refrain from sending undeliverable packets. *See* Ex. 1001 at 4:38-46 (“FIG. 4 is a flow chart that schematically illustrates a method of fault protection that is used by nodes 22 in network 20 in determining how to handle faults when they occur, in accordance with a preferred embodiment of the present invention. For the purposes of this method, each node builds and maintains two masks—a general mask and a specific mask preferably in an ordered bitmap form, with one bit for each of segments 31-37.”)

52. In the '859 Patent, each node maintains a *general mask* reflecting information about the reachability of links in the network in view of any link failures in the network and a *specific mask* reflecting the route (i.e. set of links) that, by default, packets would take from the originating node to a destination node. Link failures are detected and reported via band network fault detection communication mechanisms known in the art. Ex. 1001 at 4:38-46.

53. In order to determine the routing of a packet, the general and specific

masks of the '859 Patent are *superimposed* (i.e., the whole of one mask is compared to the whole of the other), preferably with a bitwise Boolean AND operation. Ex. 1001 at 2:31-34 (“To determine the disposition of the data flow, the node mathematically superimposes the general and specific masks, preferably by means of a Boolean conjunction operation.”), 4:64-5:1 (“In order to determine how to handle the data flow following the fault, node F mathematically superimposes the applicable general and specific masks, preferably by taking the Boolean conjunction (AND) of the masks, at a conjunction step 54”).

54. This bitwise Boolean superimposition determines the overlap (if any) of unreachable links and links in the default routing path. If there is such an overlap, then either the data should be conveyed using the alternate routing path (if the destination is reachable in the other direction) or not conveyed at all (if there are unreachable links in both directions unreachable). Ex. 1001 at 5:8-21; Cls. 3-6, 9-12.

In the situation shown in FIG. 3, however, the conjunction of the masks is 0000100, which is equal to the general mask itself. In this case, the node concludes that the current data flow overlaps all of the unreachable segments, and therefore steers the data flow around the ring in the opposite direction, at a steering step 58, as shown in the figure. On the other hand, if the conjunction is non-zero but is not equal to the general mask, the source node of the flow concludes that there are unreachable segments on both sides of the destination node, making the destination node itself unreachable. When this occurs, the source node preferably discards the packets in the data flow, at a discard step 60, in order to avoid wasting network bandwidth. Alternatively, the source node may hold the flow, and attempt to resend it later.

Ex. 1001 5:8-21.

55. The construction of the masks and the use of a bitwise Boolean superimposition operation allow for fast and efficient determination of the need to reroute or refrain from sending packets along a given path.

56. The speed afforded by bitwise Boolean operations is especially valuable in the context of packet-based networking, as a POSITA would understand that low latency—including, necessarily, low overhead of routing determinations—is highly desirable.

B. The '859 Patent's General Mask

57. The '859 Patent describes and claims the use of a “general mask” in which bits are set to indicate whether segments are reachable from a given node in the event of a fault:

1. In a bidirectional ring network, in which nodes are mutually connected by network segments, a method of fault protection, comprising:

constructing a general mask indicating **which of the segments are reachable in the event of a fault in the network by assigning a respective general mask bit to each segment, indicating whether the segment is reachable following occurrence of the fault;**

[. . .]

Ex. 1001, '859 Patent, Cl. 1.

58. Each node in the network constructs its own general mask, (Ex. 1001

4:42-45) which is desirable because in the event that a portion A of the network becomes isolated from another portion B, the nodes within the isolated portion A may be reachable to each other but not reachable to or from the remainder of the network B, and vice versa for the nodes on the other hand of the isolation divide in B.

59. A key feature of the '859 Patent's general mask is that the general mask bitmap reflects link segment *reachability* rather than link segment *failure* – i.e., it is not just a failure table. As example of how the '859 Patent's reachability table differs from merely enumerating known link segment failures is 5:32-33 of the '859 Patent specification:

If segments **33** and **35** fail, the general mask is 0011100
(since segment **34** is also unreachable).

60. In a link *failure* table, the general mask would be “0010100,” because segment 34 is not known to have failed at the time at which it becomes disconnected from the network.⁴ A link *reachability* table, however, is “0011100” because even if segment 34 remains *functional*, it is not *reachable*.

61. The difference between a link *reachability* bitmap and a link *failure*

⁴ If segment 34 failed *after* links 33 and 35, other nodes would not be aware of this due to the absence of communications with the isolated portion of the network to communicate the link failure, and a segment failure table would still (inaccurately) reflect “00010100.” A link *reachability* table, however, is necessarily accurate because it makes no claim about whether inaccessible segments are or not functional, merely that they are not reachable.

table is that bits corresponding to unreachable (but not known-to-be-failed) segments will *not* be marked in a failure table but *will* be marked in segment reachability table. This means that in any case in which two non-adjacent segments fail and two portions of the network become isolated from each other, the bits of a *link reachability* bitmap like that of the ‘859 Patent will *not be the same* as those of a bit-based link failure table. This is a key distinction between a failure table and a reachability bitmap, and one that critically differentiates the ‘859 patent from the disclosure in the asserted Omichi reference, as detailed below.

62. Thus, a bit-based failure table will not *in general* have the same values as the contents of a *reachability* bitmap or bit vector.

VIII. THE PRIOR ART - OMICHI

63. Omichi discloses a mechanism for tracking link failures in a ring network. In the event that a link failure is detected in an intended route (a “working path” around the ring, one of two possible “virtual paths” to take), Omichi discloses routing traffic to the alternative route (the other “virtual path” around the ring) based on a flowchart. Ex. 1005 at [0014].

64. The data structure that Omichi uses to track link failures is a “table” of bits, as to which link failures are marked (as either “1” in a group of “0”s or “0” in a group of “1”s) Ex, 1005 at [0036]. The values of this table are “compared” (Omichi is silent as to how) with the values of the bits of the “working path” to see if a data

flow will traverse a failed link.

[0040] Next, route selection processing part 8 determines whether or not the route switching flag that corresponds to the virtual path number that was specified in step S2 is “0” (S3). If the route switching flag is “0” (S3, 0), the used ring link table that corresponds to the virtual path number that was specified in step S2 is compared against the ring link failure table, and a determination is made as to whether or not there are bits having the same bit number and whose contents are both “1” (S4). If there are no bits having the same bit number and whose contents are both “1” (S4, No), this means either that no failure has occurred in the ring network or that even if a failure has occurred, the working path does not include a ring link where a failure has occurred. [Route selection processing part 8] then causes ring link transmission part 10 to transmit frame 5 or ATM cell 6 in the clockwise [direction] (S5).

[0041] If there are bits having the same bit number and whose contents are both “1” (S4, Yes), because this means that the working path includes a ring link where a failure has occurred, ring link transmitting unit 10 is made to transmit frame 5 or ATM-cell 6 in the counterclockwise direction. That is, frame 5 or ATM-cell 6 is transmitted using the protection path (S6). A POSITA would understand that Omichi’s disclosures differ in several key respects from the ’859 Patent’s claims.

Ex. 1005 at [0040-41].

65. Specifically, Omichi does not teach or suggest the claimed general mask, because its “link failure table” does not reflect link reachability as claimed in the ’859 Patent.

66. Additionally, Omichi’s lack of disclosure of bitwise Boolean operations, its disclosures suggestive of individual bit-by-bit comparisons in its flowcharts and specification, and the conspicuous absence of any disclosure of bitwise Boolean operations such as bitwise AND, mean that Omichi fails to teach or

suggest the use of the claimed “superimposing.”

67. Lastly, Omichi neither teaches nor suggests withholding from sending data in the event of multiple network failures making a link unreachable as claimed in dependent claims 3, 6, 9, and 12. Instead, Omichi’s disclosures reflect an assumption that this class of failure is unlikely to happen and that network conditions are unlikely to have high dynamism, and countenance only the prospect of redirection.

IX. THE CHALLENGED CLAIMS ARE NOT OBVIOUS IN VIEW OF OMICHI

A. The Petition Fails to Teach, Suggest, or Render Obvious the Claimed “General Mask”

68. Omichi fails to teach or suggest the claimed general mask of the ’859 Patent. The *link failure table* of Omichi is not the same as the claimed *segment reachability bitmap* of the ’859 Patent. The contents (i.e., bits) of the two respective data structures will differ in any instance case of a failure of non-adjacent network segments.

69. The claimed general mask of the ’859 Patent reflects segments that are *unreachable* in the event of a fault:

1. In a bidirectional ring network, in which nodes are mutually connected by network segments, a method of fault protection, comprising:

constructing a general mask indicating **which of the segments are reachable** in the event of a fault in the network by assigning a

respective general mask bit to each segment, indicating whether the segment is **reachable** following occurrence of the fault;

Ex. 1001, '859 Patent Cl. 1; (emphasis added). *See also id.* at Cl. 7 (“... the device comprising a network processor, which is adapted to **construct a general mask indicating which of the segments are reachable** in the event of a fault in the network . . .”) (emphasis added).

70. As reflected in the '859 Patent specification, in the event that two non-adjacent segments fail (separating the network into different portion), the contents of the '859 Patent's general mask will mark the intermediate segments of the disconnected network shard as unreachable:

If segments **33** and **35** fail, the general mask is **0011100** (since segment **34** is also unreachable).

Ex. 1001, '859 Patent at 5:32-33 (emphasis added).

71. This example illustrates the difference between a general mask for link *reachability* and Omichi's link *failure* table: if the '859 Patent's general mask were merely used to mark link *failures*, the general mask in this case would be “0010100,” because segment 34 is not known to have failed at the time at which it becomes disconnected. A link *reachability* table, however, is “0011100” because even if segment 34 remains *functional*, it is not *reachable*.

72. Note that if segment 34 failed *after* links 33 and 35, other nodes would not be aware of this due to the absence of communications with the isolated portion

of the network to communicate the link failure, and a segment failure table would still (inaccurately) reflect “00010100.” A link *reachability* table, however, not only differs from the contents of a link failure table in this case, but also is necessarily accurate because it makes no claim about whether inaccessible segments are functional or not, merely that they are not reachable.

1. Omichi discloses only a link failure table is not the '859 Patent's link-reachability general mask

73. Unlike the '859 Patent's general mask, Omichi discloses only a *link failure table* in which bits are set based on link failures, rather than setting bits for all link segments known to be unreachable.

[0036] FIGS. 6(a) through FIG. 6(c) show the tables that are stored in route information part 9. **The ring link failure table shown in FIG. 6(a) is used to hold information that indicates whether or not a failure has occurred in a ring link.** The ring link failure table has a bit width equal to the maximum number of ring links in the ring network, and the bit number corresponds to the ring link number. A particular bit with a value of “0” indicates that there is no failure in the corresponding ring link. A value of “1” indicates a failure in the corresponding ring link. Thus, there is no failure anywhere, and all bits in the ring link failure table are “0.” However, if a failure occurs in a ring link, a branch insertion node 1a through 1d that receives the failure notification sets the value to “1” for the bit number corresponding to the ring link number where a failure has occurred.

Ex. 1005, Omichi, at [0036] (emphasis added)

74. The link-failure data structure disclosed by Omichi is thus *not the reachability-based the general mask of the '859 Patent* and will differ in exactly the circumstances described in the '859 specification that exemplify what the general

mask is. In the ‘859 Patent, “if segments 33 and 35 fail, the general mask is **0011100** (since segment 34 is also unreachable)” (Ex. 1001, ‘859 Patent at 5 :32-33).

75. Omichi’s link failure table, by contrast, *would not mark the segments of the disconnected portion of the network* and thus *would not mark the bit corresponding to segment 34*—Omichi’s link failure table would, instead, contain the bits “0010100” – distinct from and different than the bits of the claimed general mask. Omichi’s *link failure table* is thus demonstrably *not the same* as the claimed general mask for link reachability.

2. A POSITA would not be motivated to modify Omichi to include the ‘859 Patent’s claimed general mask

76. In my opinion, a POSITA would not be motivated to modify Omichi’s link failure table to meet the claimed general mask.

77. Specifically, modifying Omichi’s link failure table to be a link reachability table provides no clear benefit to the design of Omichi: a virtual path in Omichi (i.e., a proposed path around the ring) encompasses “1” bits for each path segment. Ex 1005 at [0036] (“The ring link failure table has a bit width equal to the maximum number of ring links in the ring network, and the bit number corresponds to the ring link number. A particular bit with a value of “0” indicates that there is no failure in the corresponding ring link. A value of “1” indicates a failure in the corresponding ring link.”). Any overlap (even a single bit) between a bit corresponding to a link of a proposed virtual path and a bit of Omichi’s link failure

table indicates that the data flow cannot be sent over the proposed virtual path, and so an alternative path should be used:

[0043] Furthermore, if there are bits having the same bit number and whose contents are both “1” (S7, Yes), the working path includes a ring link where a failure has occurred. The ring link transmission part 10 is then made to transmit frame 5 or ATM cell 6 in the clockwise direction. In other words, frame 5 or ATM cell 6 is sent using the protection path.

Ex. 1005 at [0043]. *See also id.* at [0040-42].

78. Thus there is no need for a POSITA to (redundantly) ensure that each bit of the failure table corresponding to a network segment isolated by multiple non-adjacent faults should be set to “1,” because any single match between a bit index of link failure table and the corresponding bit index of the virtual path of Omichi (i.e. the “used ring link table,” *See* Ex. 1005 at [0037], Fig. 6b), would be sufficient on its own to enable Omichi’s design.

79. Thus, there is no motivation for a POSITA to modify Omichi’s link failure table to meet the requirements of the ’859 Patent’s claimed link-reachability general mask, because such a modification does not improve the design or function of Omichi.

80. Because the claimed general mask is an element of the two independent claims of the ’859 Patent, and thus of every element of the ’859 Patent, in my opinion Omichi does not teach, suggest, or render obvious the claimed general mask of the ’859 Patent.

B. Omichi Fails to Teach, Suggest, or Render Obvious the Claimed “Superimposing”

81. In my opinion, a POSITA would not understand Omichi to disclose or render obvious the that the claimed general mask and specific masks are “superimposed” as claimed in the ’859 Patent. *See* Ex 1001 at Cl. 1 (“superimposing the general and specific masks in order to determine a disposition of the flow.”), Cl. 7 (“superimpose the general and specific masks in order to determine a disposition of the flow.”).

82. As I discussed above with respect to claim construction, a POSITA would understand “superimposing” in the ’859 Patent to mean “as part of a routing determination, combining or comparing by overlaying all bits as a whole.”

1. Only Bitwise Boolean operations, rather than serial logical Boolean operations, would meet the “superimposing” claim element

83. In order for the claimed general and specific masks to be “superimposed,” a POSITA would understand that the bits of the respective masks must be *overlaid* (i.e., compared or combined in their entirety) – generally by means of a bitwise binary⁵ Boolean operation such as bitwise AND (in computer programming, typically represented by the ‘&’ operator), bitwise OR (‘|’), or bitwise XOR (‘^’). (*See, e.g.*, Ex. 1001 Cl. 2 (“2. A method according to claim 1, wherein

⁵ “Binary” here referring to taking two arguments rather than to operating on zeroes and ones.

superimposing the masks comprises combining the masks by means of a Boolean operation.”) *See also*, e.g., Petition at 1 (“To determine if data for that path should be rerouted in the opposite direction, the first set of bits is compared with the second set of bits using a Boolean AND operation “) (citing Ex. 1001 at 4:64-5:5). *See also*, e.g. Ex. 1001 at 4:64-67 (“In order to determine how to handle the data flow following the fault, node F mathematically superimposes the applicable general and specific masks, preferably by taking the Boolean conjunction (AND) of the masks[.]”).

84. A POSITA would understand that taking the Boolean conjunction (AND) of the masks (which are described as “bitmaps” 2:23-30; Cl. 1, Cl. 7) to yield a new bit vector (such as “00000000” or “0000100” as discussed as Ex. 1001 at 4:64-5:13) refers to a *bitwise* binary operation, typically represented in source code by the operator “&” (single ampersand) and taking as arguments unsigned integers.

85. Bitwise Boolean operations differ somewhat conceptually (and, in many languages, such as Java, accept different argument types) from *Logical* Boolean operations. Logical Boolean operations work conceptually on propositions that are “true” or “false” (such as “is X greater than 10”), and have an associated primitive “boolean” (e.g. Java) or “bool” type (e.g. C#, Python) that can take the values “true” or “false”) in more modern languages such as Java and Python (which were initially released in the 1990s) and C# (released around or slightly after the

invention of the '859 Patent).

86. By contrast, *bitwise* Boolean operators take as inputs and provide as outputs *sequences of bits* rather than *logical propositions*, and are more akin to mathematical or even physical operators such as arithmetic operators like “+”, “-“, etc..

87. The distinction between logical and bitwise operators is reflected at a source code level: logical Boolean operations typically used different or doubled characters (“&&” for AND, “||” for OR, “!” for NOT) whereas *bitwise* operators use single ones (“&” for bitwise AND, “|” for bitwise OR, “~” for bitwise NOT, “^” for bitwise XOR).

88. While older languages such as C can technically evaluate logical propositions for numeric types (e.g. “zero” standing in for false and “nonzero” standing in for true), as reflected in the trend of language development this is in some ways a concession to weaker typing systems and earlier developmental paradigms. More modern languages tend to use their type systems to enforce the conceptual distinction between logical and bitwise operations at code-level.

89. A POSITA would understand that using bitwise operations for was a typical use of “masks” in network operations and had the advantage of high speed and efficiency in a context (packet routing) where low latency was at a premium.

90. As would be understood by a POSITA, bitwise operations in the

computer programming context, such as bitwise AND,⁶ are low-level operations whose invocation in source code closely parallels the implementation of such operations down to the register- or logic gate-level when compiled into machine code. As such, bitwise comparisons such as bitwise AND tend to be extremely fast operations that require few instructions to implement when the source code is compiled and that are executed very quickly.

91. As would be understood by a POSITA, fast execution is at a premium in a networking environment when making packet-routing determinations, because each such determination adds latency in a context in which minimal latency is highly desirable. *See, e.g.* 1001 ('859 Patent) at 1:1-6 (“Field of the Invention” section) “The present invention relates generally to communication networks, and specifically to high-speed packet rings); 1:9-14 (“Network ring topologies are gaining in popularity, particularly in Internet Protocol (IP) networks. Such networks enable carriers to offer large bandwidth to users in a cost-effective manner. They also lend themselves to fast rerouting in the event of network failures . . . “).

92. An advantage of the '859 Patent – as reflected in its title, “Fast Protection in Ring Topologies” – is the use of such bitwise superimposition operations to compare the intended packet route reflected in the specific mask with

⁶ By convention, Boolean operations (*e.g.* AND, OR, NOT, XOR) are capitalized in the programming arts.

the reachability table reflected in the general mask, because such operations tend to be extremely fast and incur relatively little computational overhead. Because of the minimal computational overhead, such operations are well-suited to environments requiring minimal latency (such as network routing comparisons) and minimize the overhead of the fault-sensitive routing decisions and path selection claimed in the '859 Patent.

93. The Petition and Dr. Min's declaration rely on the alleged disclosure of Boolean operations in Omichi in support of the Petition. *E.g.* Petition at 1 ("To determine if data for that path should be rerouted in the opposite direction, the first set of bits is compared with the second set of bits using a Boolean AND operation") (citing Ex. 1001 at 4:64-5:5) ("In order to determine how to handle the data flow following the fault, node F mathematically Superimposes the applicable general and specific masks, preferably by taking the Boolean conjunction (AND) of the masks, at a conjunction step 54. If the conjunction is 0000000, the node can conclude that there is no overlap between this data flow and the failed segments in network 20, and therefore determines that no steering of this flow is required, at a non-action step 56."). Ex. 1003 (Min Petition) at ¶ 46 (discussing construction of "mask") ("If one were to combine (e.g., logically AND) this mask with another data item, the result of that combination (e.g., AND operation) would retain only the last two bits. The first six bits would be eliminated. In particular, if the mask 00000011 is combined

with the value 10101010, the result would be 00000010.”), ¶¶ 48-50.

94. However, the Petition and Dr. Min’s declaration do not, in my opinion clearly distinguish between serial, bit-by-bit *logical* Boolean operations (for example, the conjunction of the propositions “bit 12 of the general mask is 1” and “bit 12 of the specific mask is 1”, applied to each of the bits in the respective bit vectors) versus a *bitwise* Boolean comparison of two bit vectors as a whole. Such a distinction is important, however, because only a *bitwise* Boolean comparison would meet the claimed superimposition of the ’859 Patent.

95. Accordingly, whether Omichi renders the claimed superimposition obvious depends on whether it specifically renders *bitwise* Boolean operations specifically obvious.

2. The claimed “superimposition” (i.e. bitwise Boolean operations) are not taught or suggested by Omichi and instead such disclosures are conspicuously absent

96. Unlike the ’859 Patent’s disclosure of Boolean AND operations to yield a bitmap (which a POSITA would understand to reflect the use of bitwise Boolean AND, *See, e.g.* Ex. 1001 at 5:25-28 (“If node E fails, then the general mask is 0000011. The conjunction of this mask with the specific mask 0001111 is 000011 (equal to the general mask), and the data flow is steered as in the case shown in FIG. 3.”), Omichi refers only to “comparing” the “tables” of route information Ex. 1005 at [0040]. While the Petition argues that “Omichi’s comparing of both the ring link

failure table and the used ring link table is within the scope of “superimposing” as recited in claim 1” (Petition at 35), I believe this overlooks what a POSITA would infer from viewing Omichi.

97. This are any number of ways by which comparisons of bit-valued data may be accomplished that do not involve superimposition. A POSITA would understand that the recited “comparison” of Omichi would not necessarily imply the use of bitwise binary operations that would practice the claimed superimposition, and instead could refer to *logical* Boolean operations (represented by different operators in typical programming languages – i.e., ‘&&’ for logical AND versus a single ampersand ‘&’ for *bitwise* AND) performed bit-by-bit, without any overlay of Omichi’s bit-tables as a whole. Such a comparison could cease as soon as a desired virtual pathway was found to have a link failure, such that not even every pair of bits would be compared, let alone compared as a whole.

98. For example, the following code (in a language such as C or C#) compares the “tables” of Omichi bit-by-bit without superimposing them or using a bitwise Boolean operator (such as AND (“&”) or OR (“|”))⁷, and if implemented would facilitate comparing bits for equality while also making clear precisely which bit of the general and specific masks are both “1” (i.e., which link has both failed

⁷ The example code does use the bitwise “bit shift” operator “>>” but this is not a Boolean operation, instead being mathematically equivalent to dividing by 2 *n* times in binary integer arithmetic.

and is in the proposed working path of Omichi):

```
for(int i = 0; i < Omichi_table_length; i++)
{
    uint link_failure_status_bit =
        (ring_link_failure_table >> i) % 2;
    uint desired_working_path_bit =
        (used_ring_link_table >> i) % 2;
    if (link_failure_status_bit == 1 &&
        desired_working_path_bit == 1)
    {
        // desired path has a link failure at
        // bit "i" - take appropriate action
        // and stop comparison

        use_alternative_path();

        break;
    }
}
```

99. Accordingly, contrary to what I understand to be the position of the Petition and Dr. Min, in my opinion Omichi does not teach the claimed superimposition explicitly, nor is such superimposition necessarily present in Omichi.

100. Omichi does not clearly state the use of bitwise Boolean operations in its disclosures. Omichi instead refers to the “comparison” of its link failure tables and working paths without any description of a specific mechanism:

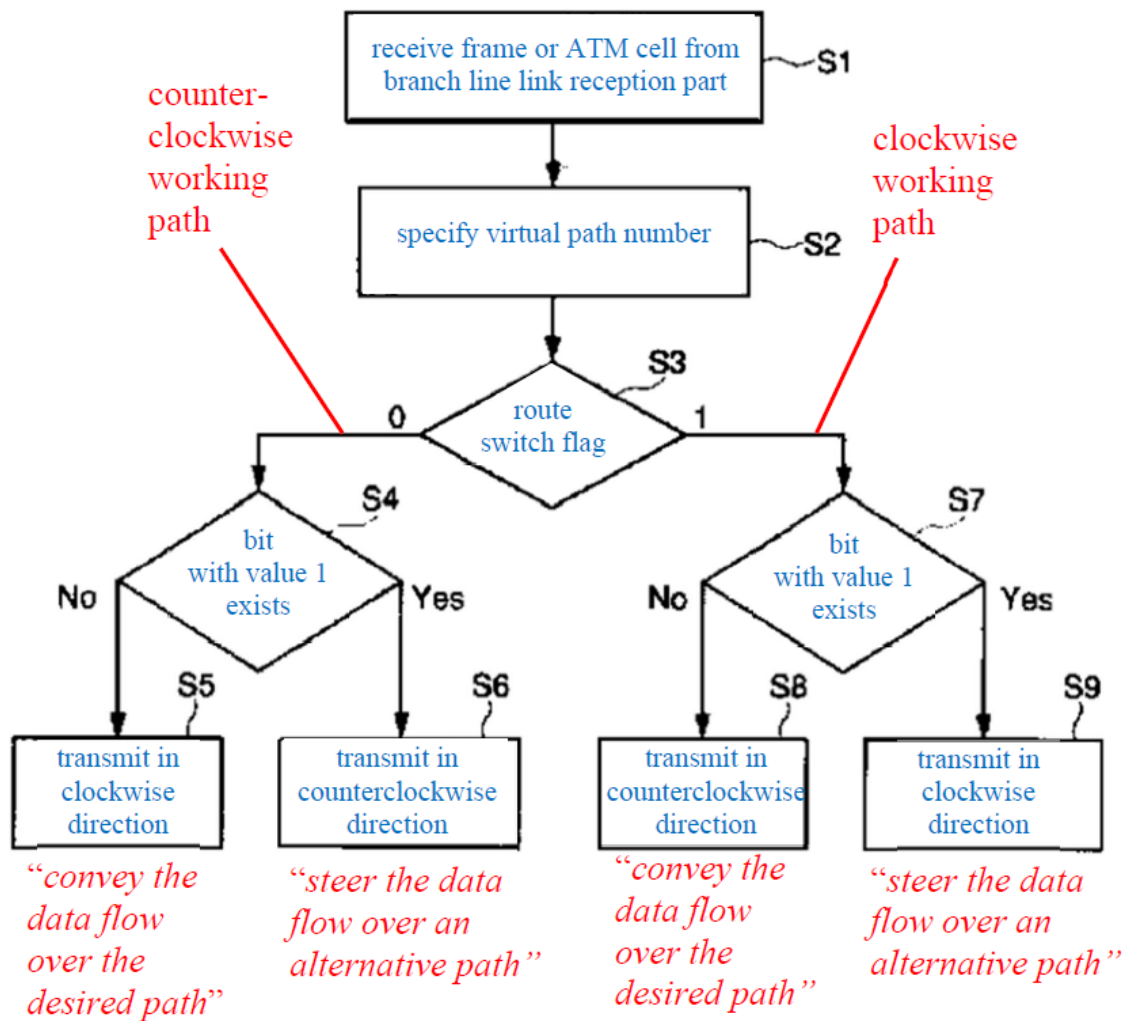
[0040] Next, route selection processing part 8 determines whether or not the route switching flag that corresponds to the virtual path number that was specified in step S2 is “0” (S3). If the route switching flag is “0” (S3, 0), the used ring link table that corresponds to the virtual path number that was specified in step S2 **is compared against** the ring link failure table, **and a determination is made** as to whether or not there are bits having the same bit number and whose contents are both “1” (S4).

Ex. 1005 at [0040] (emphasis added).

101. On its own, the use of this language suggests the potential use of logical Boolean operations—the phrases “compared against” and “determination is made” would be understood to suggest logical Boolean operations by default.

102. Further supporting this inference, Omichi refers to its link failure data structure as a “table,” which would typically be understood by a POSITA to suggest a data structure with individually addressable or indexable entries (for example, an array) in which entries are compared element-by-element rather than as, e.g., arguments to a binary bitwise operation.

103. At least one flowchart in Omichi also appears to indicate that a specific bit-valued quantity should be evaluated on its own when making routing determinations – the “switch route flag” S3 in Figure 7



Omichi, Fig. 7 (annotated).

(Excerpted from Min Decl. ¶ 98).

104. Dr. Min suggests that “Boolean logic would have been within the background knowledge of a POSITA” (Ex. 1003, Min Decl. at ¶ 35), seemingly referring to both bitwise and logical Boolean operations (*id.* at ¶¶ 32-34). I disagree with Dr. Min’s implicit conflation of the two concepts (bitwise and logical Boolean operations) in the programming arts as discussed above, as I believe the distinction

to be relevant to whether the recited “comparison” suggests the claimed superimposition or not.

105. While I disagree with the conflation of bitwise and logical operations, I do agree with Dr. Min that both such types of operations would have been known to a POSITA at the time of the invention of the ‘859 Patent.

106. However, in my experience, when *bitwise* Boolean operations are being employed they are most usually described explicitly (or else in conjunction with terminology such as “masks” that are understood to implicate the use of bitwise operations), in part because the results of such operations on bit vectors are typically less intuitive than those of logical Boolean operations.

107. Accordingly, were bitwise Boolean comparisons present and employed in Omichi (or any other reference), a POSITA would expect to see the nature of such a bitwise Boolean operation explicitly discussed. The conspicuous absence of any such discussion, in a context in which a POSITA might expect to see it referenced if it were present (i.e. comparisons of two bit-valued data structures (*See* Omichi FIGS. 6a, 6b)), gives rise to at least potential inference or suggestion that Omichi does not contemplate the use of bitwise Boolean operators, relying instead perhaps on element-by-element comparisons.

108. Accordingly, I do not believe that Omichi clearly teaches or suggests the claimed superimposing operation of the ‘859 Patent (*i.e.* the bitwise Boolean

operations relied on by the Petition and Dr. Min) to a POSITA.

C. Omichi Does not Disclose or Render Obvious not Conveying a Data Flow when the Destination Node is Unreachable [Claims 3, 6, 9, 12]

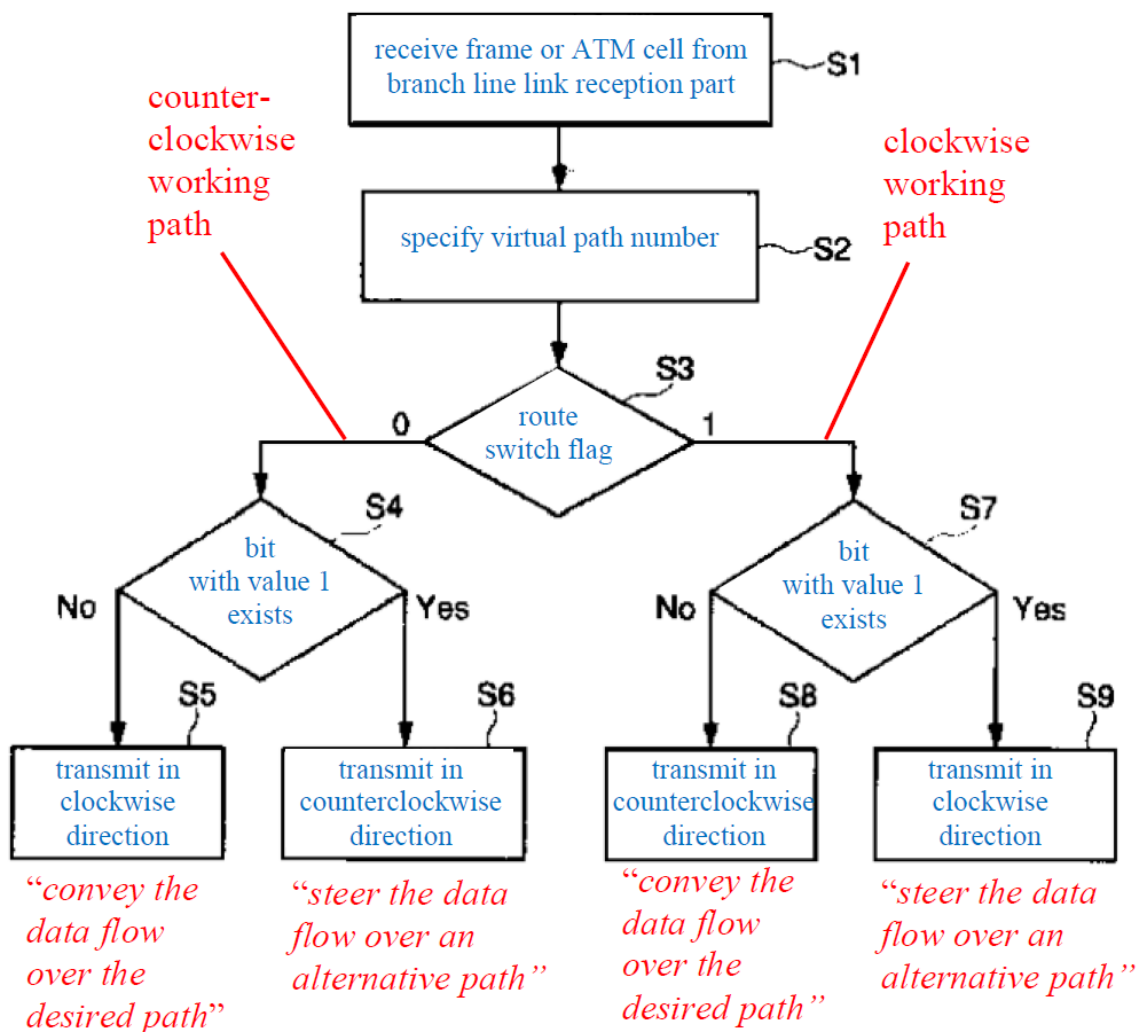
109. In my opinion, Omichi does not teach or suggest to a POSITA ceasing to provide a data flow (i.e., not sending packet) when a destination node or segment is unreachable, which forms a part of ‘859 Patent Claims 3 (*A method according to claim 1, wherein superimposing the masks comprises determining whether to convey the data flow over the desired path, to steer the data flow over an alternative path, or to stop conveying the data flow*“) 6 (*“A method according to claim 3, wherein superimposing the masks comprises generating a combined mask, and determining that the data flow should not be conveyed when the combined mask is neither null, nor does it correspond to the general mask.”*), 9 (parallel in relevant part to claim 3)⁸ (See Petition 9)), and 12 (parallel in relevant part to claim 6)⁹

110. Omichi clearly teaches a fundamentally binary view of data transmission (i.e., one path or the other around the ring), as reflected by the flowchart

⁸ “A device according to claim 7, wherein the processor is adapted to determine whether to convey the data flow over the desired path, to steer the data flow over an alternative path, or to stop conveying the data flow based on the superimposed masks.”

⁹ “A device according to claim 9, wherein the processor is adapted to superimpose the masks so as to generate a combined mask, and to determine that the data flow should not be conveyed when the combined mask is neither null, nor does it correspond to the general mask.”

of Fig. 7 (See, e.g. elements S3, S5, S6, S8, S9) and does not contemplate the multiple-failure scenario in which failures exist on both sides of the ring network.



Ex.1005, Fig. 7 (annotated); Ex.1003, ¶98.

(Fig. 7 (annotated), reproduced, including annotations, from Petition at 45).

111. Omichi thus implicitly reflects an underlying assumption that multiple non-adjacent link failures are unlikely to isolate a network segment.

112. This is also consistent with Omichi at [0044], which appears to

contemplate¹⁰ the resetting of alternative paths back to their original working paths only on the clearing of all failures in the network, rather than dynamically switching from the alternate “protection paths” back to the default “working paths” as individual segments recover from failure.

[0044] When a ring link recovers from a failure, the branch insertion node that detects the recovery from the failure notifies the other branch insertion nodes of the recovery from the failure. The branch insertion nodes that are notified of the recovery from the failure then set the corresponding bit in the ring link failure tables to “0.” **When all bits in the ring link failure tables are cleared, the virtual paths that were replaced by protection paths are all switched back to using working paths so that the ring network is operated in a state with no failures.**

Ex. 1005 [0044].

113. The disclosures of Omichi reflect a set of assumptions of infrequent network failures typically fixed all at once, rather than dynamic network conditions requiring more robust failure modes. In view of these assumptions, it would not be obvious to a POSITA to modify (or try to modify) Omichi to account for a situation that was not expected to arise.

114. Because reflects a set of assumptions that network failures unlikely to occur on both potential paths of a ring network, a POSITA viewing Omichi’s binary

¹⁰ [0044] Is not entirely clear as to whether it is making a descriptive rather than a prescriptive statement, but a descriptive statement would appear to be at least somewhat redundant or tautological – essentially akin to stating that “each path is reset to its default path when links have been restored, so when there are no errors left in the network, all data will traverse its default paths.”

path-decision flowchart and related disclosures would not understand it to teach or suggest refraining from sending data in the event of multiple network failures resulting in complete disconnection, nor would a POSITA be motivated to modify Omichi to include such behavior in view of such assumptions.

X. CONCLUSION

115. For the foregoing reasons, I do not believe that Omichi renders the ‘859 Patent obvious to a POSITA.

116. This declaration and my opinions herein are made to the best of my knowledge and understanding, and based on the material available to me, at the time of signing this declaration. I declare that all statements made herein on my own knowledge are true and that all statements made on information and belief are believed to be true, and further, that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 or Title 18 of the United States Code.

Date: September 25, 2025

Respectfully submitted,



Nenad Medvidović